Robustifying the Flock of Trackers

Tomáš Vojíř and Jiří Matas
{matas, vojirtom}@cmp.felk.cvut.cz

Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague
Outline

• The Flock of Trackers (FoT) class of trackers
• The Median-flow FoT
• Local tracker placement in FoT
• Local tracker filters
• Results
• Conclusions
The Flock of Trackers (FoT)

- Transformation is estimated from a number of independent local trackers (which provide frame-to-frame correspondences)

- Local tracker positions are a parameter of the method

- Quality of estimation depends on
  1) Local tracker placement
  2) Local tracker method
  3) Model and Estimator of the global motion
Local trackers

- Any algorithm which establishes correspondences of selected patches (points of interest) between two images and satisfies:
  1) May track any local regions
  2) High computation speed (be able to run multiple instances at once, e.g. 100)

Note: we (and also Kalal et al.\cite{1}) use the Pyramidal implementation of the Lucas-Kanade\cite{2} feature tracker.

\begin{itemize}
\end{itemize}
F-B filtered Median-Flow

- Local trackers placed on a regular grid \( \equiv \text{Grid FoT} \)
- Local trackers filtered by normalized cross-correlation and so the called Forward-Backward procedure (the filtered set is denoted \( F_s \))
- Estimator = Median (from \( F_s \))
- Transformation is modelled as translation and scale
- Theoretically robust up to 50% of outliers for translation and \( 100 \times (1-\sqrt{0.5})=29\% \) for scale (estimated from pairs of correspondences) in \( F_s \)

Local tracker placement in Grid FoT

- **Standard approach:**
  - Uniformly cover the object of interest (or select good-point-to-track)
  - Re-init trackers after failure to default position (or find a new suitable good-point-to-track)

- **We present a novel local tracker placement => Cell FoT**
  - Cell = a region of local tracker “free movement”
  - Store the offset to cell center for each local tracker
  - Reinitialize out-of-cell local tracker positions to cell centers
Local tracker placement in Cell FoT

- Idea:
  "good points to track are those the trackers drifts to"
  -> Novel approach to selecting good-point-to-track

- The object is still uniformly covered
  => robustness to occlusion and pose changes holds

- We experimentally showed that the Cell FoT is superior to Grid FoT
Local trackers filter

- Filtering methods used in F-B Median-flow:
  1) Normalized cross-correlation (shown to be superior to SSD in local patch filtering in FoT tracking problems)
  2) Forward-Backward ("reverse tracking")

- Both are ranking filters
- Forward-Backward is expensive – almost slowing tracking 2x (processing time grows by \( \approx 72\% \))

- We present two new filters and combined them with the NCC filter
- Result: a superior Local trackers filter, denoted \( \Sigma \)
Outlier filters – NCC, F-B

NCC
- Compute normalized cross-correlation between local tracker patch in time $t$ and $t+1$
- Sort local trackers according to NCC response
- Filter out bottom 50% (Median)

Forward-Backward\textsuperscript{1}
- Compute correspondences of local trackers from time $t$ to $t+k$ and $t+k$ to $t$ and measure the $k$-step error
- Sort local trackers according to the $k$-step error
- Filter out bottom 50% (Median)

\textsuperscript{1} Z. Kalal, K. Mikolajczyk, and J. Matas. Forward-Backward Error: Automatic Detection of Tracking Failures. ICPR, 2010
The combined outlier filter Σ

- Combining three types of information:
  a) Local appearance (NCC)
  b) Spatial consistency (Nh) (similar to smoothness assumption used in optic flow estimation)
  c) Temporal consistency (MMp)

- Together form very a strong filter

- Negligible computational cost (less than 10%)
Outlier filters – MMp

- MMp models local trackers as two states (i.e. inlier, outlier) probabilistic automaton with transition probabilities $p^i(s_{t+1} | s_t)$

- MMp compute probability of being inlier for all local trackers -> filter by
  1) Static threshold $\Theta_s$
  2) Random threshold $\Theta_r$

- Learning is done incrementally (learns are the transition probabilities between states)

- Can be extended by “forgetting”, which allows faster response to object appearance change
Outlier filters – \( N_h \)

- For each local tracker \( i \) is computed neighbourhood consistency score as follows:

\[
S_i^{N_h} = \sum_{j \in N_i} \left[ \| \Delta_j - \Delta_i \| < \varepsilon \right], \quad \text{where} \quad [expression] = \begin{cases} 
1 & \text{if expression is true} \\
0 & \text{otherwise}
\end{cases}
\]

\( N_i \) is four neighbourhood of local tracker \( i \), \( \Delta \) is displacement and \( \varepsilon \) is displacement error threshold.

- Local trackers with \( S_i^{N_h} < \Theta_{N_h} \) are filtered out.

- Setting:
  \( \varepsilon = 0.5 \text{px} \)
  \( \Theta_{N_h} = 1 \)
## Results on publish sequences

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>17</td>
<td>n/a</td>
<td>94</td>
<td>135</td>
<td>761</td>
<td>761</td>
</tr>
<tr>
<td>2</td>
<td>75</td>
<td>313</td>
<td>44</td>
<td>313</td>
<td>170</td>
<td>76</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>6</td>
<td>22</td>
<td>101</td>
<td>140</td>
<td>140</td>
</tr>
<tr>
<td>4</td>
<td>33</td>
<td>8</td>
<td>118</td>
<td>37</td>
<td>97</td>
<td>264</td>
</tr>
<tr>
<td>5</td>
<td>50</td>
<td>5</td>
<td>53</td>
<td>49</td>
<td>52</td>
<td>52</td>
</tr>
<tr>
<td>6</td>
<td>163</td>
<td>n/a</td>
<td>10</td>
<td>45</td>
<td>510</td>
<td>510</td>
</tr>
</tbody>
</table>

- Result for other algorithms obtained from [1]
- Best achieved results for sequences are marked bold

---

Results - robustness

FoT – NCC, FB

FoT – Σ
Conclusion

• Contributions:
  1. Structural improvement in Grid FoT scheme
  2. Two new local trackers filters

• A new suggestion for the “good-point-to-track” problem
  (≈the points trackers drifts to)

• MMp can be indirectly performs on-the-fly motion segmentation
Live Demo